MICRO FOCUS

Open Enterprise Server 2023

Storage Services™ Auditing Client Logger (VLOG) Utility Reference

October 2022

Legal Notices

© Copyright 2022 Micro Focus or one of its affiliates.

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Contents

About This Guide 5

1 Overview of the NSS Auditing Client Logger (VLOG) Utility 7

1.1 Using VLOG with the NSS Auditing Engine . . . 7
- 1.1.1 Logged Output . . . 7
- 1.1.2 Paths to Include or Exclude . . . 7
- 1.1.3 File System Events to Monitor . . . 7
- 1.1.4 NSS, NCP, and CIFS Event Sub-Types to Monitor . . . 8
- 1.1.5 VIGIL Events to Monitor . . . 8

1.2 Using Auditing Client Applications with the NSS Auditing Engine . . . 8
- 1.2.1 NetIQ Sentinel Server . . . 9
- 1.2.2 Third-Party Partner Applications . . . 9
- 1.2.3 ArcSight Enterprise Security Manager (ESM) . . . 9

2 What's New or Changed in the NSS Auditing Client Logger (VLOG) Utility 11

2.1 What's New or Changed in VLOG (OES 2023) . . . 11

3 Installing the NSS Auditing Engine and Client Logger 13

4 Running VLOG in a Virtualized Environment 15

5 Running VLOG Along with ArcSight Smart Connector 17

5.1 Syslog Pipe . . . 17

5.2 ArcSight Common Event Format (CEF) Folder Follower Scanner . . . 17

6 VLOG Utility Man Page 19

vlog . . . 20

7 Common Event Format (CEF) 43

About This Guide

This reference guide describes the syntax and options for the Storage Services (NSS) Auditing Client Logger (VLOG) utility for Open Enterprise Server (OES) 2023. The VLOG utility is used with the NSS Auditing Engine.

This guide includes the following sections:

- Chapter 1, “Overview of the NSS Auditing Client Logger (VLOG) Utility,” on page 7
- Chapter 2, “What’s New or Changed in the NSS Auditing Client Logger (VLOG) Utility,” on page 11
- Chapter 3, “Installing the NSS Auditing Engine and Client Logger,” on page 13
- Chapter 4, “Running VLOG in a Virtualized Environment,” on page 15
- Chapter 5, “Running VLOG Along with ArcSight Smart Connector,” on page 17
- Chapter 6, “VLOG Utility Man Page,” on page 19
- Chapter 7, “Common Event Format (CEF),” on page 43

Audience

This guide is intended for system administrators or anyone who is responsible for auditing file system events on NSS file systems on OES servers.

Knowledge of the NSS file system is assumed. Some background knowledge of the host operating system is also assumed.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation.

Documentation Updates

The VLOG man page, `vlog(8)`, is available on the server. Updates to the man page are delivered with any updates to the VLOG utility.

For the most recent version of the *NSS Auditing Client Logger (VLOG) Reference*, visit the Open Enterprise Server 2023 Documentation Web site (https://www.microfocus.com/documentation/open-enterprise-server/2023/).

Additional Documentation

Information about the NSS Auditing Engine SDK (Software Development Kit) is available on the NSS Auditing SDK Web site (http://www.novell.com/developer/ndk/nss_auditing_sdk.html).

1 Overview of the NSS Auditing Client Logger (VLOG) Utility

The Storage Services (NSS) Auditing Client Logger (VLOG) utility for Open Enterprise Server (OES) is used with the NSS Auditing Engine (`/usr/lib/systemd/system/novell-vigil.service`).

- Section 1.1, “Using VLOG with the NSS Auditing Engine,” on page 7
- Section 1.2, “Using Auditing Client Applications with the NSS Auditing Engine,” on page 8

1.1 Using VLOG with the NSS Auditing Engine

When VLOG is running, it intercepts, parses, filters, augments, and displays auditing records received from the NSS Auditing Engine (`vigil`). For information about configuring and using the VLOG utility, see Chapter 6, “VLOG Utility Man Page,” on page 19.

The basic functionality includes:

- Section 1.1.1, “Logged Output,” on page 7
- Section 1.1.2, “Paths to Include or Exclude,” on page 7
- Section 1.1.3, “File System Events to Monitor,” on page 7
- Section 1.1.4, “NSS, NCP, and CIFS Event Sub-Types to Monitor,” on page 8
- Section 1.1.5, “VIGIL Events to Monitor,” on page 8

1.1.1 Logged Output

By default, `vlog` sends its output to `stdout` in an XML record format. VLOG also supports output in CSV (comma-separated values) format, SENT format (for NetIQ Sentinel/Log Manager products), and CEF (Common Event Format). For information, see “VLOG Options” on page 21.

1.1.2 Paths to Include or Exclude

VLOG allows you to specify which files and directories are to be monitored. You can specify patterns for the file and directory names by using a defined set of search characters. You can specify which file paths are to be included or excluded. For information, see “Path Element Options” on page 29. For examples of path patterns, see “Path Element Examples” on page 30.

1.1.3 File System Events to Monitor

VLOG can be configured to log various file system events on files and directories that are reported by the NSS Auditing Engine, including:

- delete

- create
- open
- close
- rename
- link
- metadata modified
- trustee added or removed
- inherited rights modified

For information, see “Event Types” on page 36 and “Event Type Examples” on page 37.

1.1.4 NSS, NCP, and CIFS Event Sub-Types to Monitor

These NSS file system events can be audited by NSS, NCP (NetWare Core Protocol), and CIFS sub-types. For information, see “Event Sub-Types NSS, NCP, and CIFS” on page 37 and “Event Sub-Type Examples” on page 38.

1.1.5 VIGIL Events to Monitor

VLOG can also be configured to report various events internal to the NSS Auditing Engine, referred to as VIGIL events, such as:

- Starting or stopping the `vigil.ko` kernel module
- Starting or stopping the `vigil.ncp.ko` kernel module
- Starting or stopping the `vigil.nss.ko` kernel module
- Starting or stopping the `vigil.cifs.ko` kernel module
- Starting or stopping the Auditing Client (an internal construct of the NSS Auditing Engine)
- Starting or stopping the Auditing Client User (an internal construct of the NSS Auditing Engine)
- Rolling the audit record log file over to a new file when the log reaches an administrator-specified maximum size

For information, see “Patterns for Filtering Records of Type VIGIL” on page 25 and “Examples for Filtering VIGIL Events” on page 27.

1.2 Using Auditing Client Applications with the NSS Auditing Engine

Some auditing client applications, such as NetIQ Sentinel and various third-party products, can access audited events that are reported by the NSS Auditing Engine. Information about the NSS Auditing Engine Software Developer Kit (SDK) is available on the NSS Auditing SDK Web site (http://developer.novell.com/wiki/index.php/NSS_Auditing_SDK).

- Section 1.2.1, “NetIQ Sentinel Server,” on page 9
- Section 1.2.2, “Third-Party Partner Applications,” on page 9
- Section 1.2.3, “ArcSight Enterprise Security Manager (ESM),” on page 9

1.2.1 NetIQ Sentinel Server

NetIQ Sentinel server can be used to collect and report on event logs from the NSS Auditing Client Logger utility. NetIQ Sentinel server runs on a 64-bit SUSE Linux Enterprise Server (SLES) 12 host. You can download NetIQ Sentinel server from the Micro Focus Download Web site (https://download.novell.com/protected/Summary.jsp?buildid=eVXYzszOkxo~). A 60-day evaluation license is available on the download site. For installation and usage instructions, see the Sentinel Documentation Web site (https://www.netiq.com/documentation/sentinel-74/).

1.2.2 Third-Party Partner Applications

The following Micro Focus partners are developing applications for use with the NSS Auditing Engine:

- Blue Lance
- NetVision
- Symantec

1.2.3 ArcSight Enterprise Security Manager (ESM)

The ArcSight ESM is a centralized system for logging, analyzing, and managing events from different sources. The OES server can forward events to ArcSight ESM with the help of ArcSight smart connector. Information about ArcSight ESM is available on the MicroFocus Web site (https://www.microfocus.com/en-us/cyberres/secops/arcsight-esm). Information about Smart Connector is available on the ArcSight Marketplace (https://marketplace.microfocus.com/arcsight/content/v2/connector).

2 What's New or Changed in the NSS Auditing Client Logger (VLOG) Utility

This section describes the changes and enhancements for the Storage Services (NSS) Auditing Client Logger (VLOG) Utility since the release of Open Enterprise Server (OES) 2023.

- Section 2.1, "What's New or Changed in VLOG (OES 2023)," on page 11

2.1 What's New or Changed in VLOG (OES 2023)

VLOG in OES 2023 has been modified for bug fixes. There are no new features or enhancements in OES 2023.

3 Installing the NSS Auditing Engine and Client Logger

The Storage Services (NSS) Auditing Client Logger (VLOG) utility for Open Enterprise Server (OES) is used with the NSS Auditing Engine (`/usr/lib/systemd/system/novell-vigil.service`). The Vigil engine and VLOG utility are installed by default when you install the **OES Storage Services** pattern on your OES server. The following packages are installed:

- novell-vigil
- novell-vigil-libs
- novell-vigil-vlog

For information about installing, upgrading, or patching OES Services on an OES server, see the OES 2023: Installation Guide.

For information about installing or using the NSS file system, see the OES 2023: NSS File System Administration Guide for Linux.

4 Running VLOG in a Virtualized Environment

The Storage Services Auditing Client Logger (VLOG) utility runs in a virtualized environment just as it does on a physical server running Open Enterprise Server, and requires no special configuration or other changes.

- For information on setting up virtualized OES server, see "Installing, Upgrading, or Updating OES on a VM" in the OES 2023: Installation Guide.
- To get started with Xen virtualization and KVM virtualization, see the Virtualization Guide (https://documentation.suse.com/sles/15-SP4/html/SLES-all/book-virtualization.html).
- To get started with third-party virtualization platforms, such as Hyper-V from Microsoft and the different VMware product offerings, refer to the documentation for the product you are using.

5 Running VLOG Along with ArcSight Smart Connector

The ArcSight ESM system is a centralized system for logging, analyzing, and managing events from different sources. The OES server can forward events to ArcSight ESM with the help of ArcSight smart connector.

The Storage Services Auditing Client Logger (VLOG) output can be integrated with ArcSight using any of the following ArcSight smart connector type:

- Section 5.1, “Syslog Pipe,” on page 17
- Section 5.2, “ArcSight Common Event Format (CEF) Folder Follower Scanner,” on page 17

5.1 Syslog Pipe

Perform the following task to integrate VLOG with ArcSight using Syslog Pipe smart connector type:

1. Create a named pipe by executing the “`mkfifo`” command. For example, `mkfifo /var/tmp/syspipe`.
2. Configure smart connector with type “`Syslog Pipe`” and named pipe absolute path, `/var/tmp/syspipe`.
3. Run the vlog utility with the option, `/opt/novell/vigil/bin/vlog -d -f cef -o /var/tmp/syspipe`.

5.2 ArcSight Common Event Format (CEF) Folder Follower Scanner

Perform the following task to integrate VLOG with ArcSight using ArcSight CEF folder follower scanner smart connector type:

1. Configure the smart connector to listen on vlog output directory. For example, `/root/vlogs`.
2. Run vlog utility with the option, `/opt/novell/vigil/bin/vlog -d -f cef -R 100MB -o /root/vlogs/vlog_cef_output.cef -t`.

   NOTE: For the ArcSight smart connector folder follower scanner to process the `*.cef files`, we have to create a trigger file with the name `*.cef_ready`. The “`-t`” option is introduced to do this automatically when provided along with format type CEF. For other formats, an option will create the trigger file without any name. If needed, we can provide any custom extension as the argument to the “`-t` “option.

3. Log files once processed will be renamed as, `vlog_cef_output<timestamp>.cef_processed`.

6 VLOG Utility Man Page

This section provides the syntax, options, and examples for the Storage Services (NSS) Auditing Client Logger (VLOG) utility for Open Enterprise Server (OES). This information is also available on the server as the `vlog(8)` man page.

- "Synopsis" on page 20
- "Availability" on page 20
- "Syntax" on page 20
- "Description" on page 21
- "VLOG Options" on page 21
- "Filtering Records" on page 25
- "Patterns for Filtering Records of Type VIGIL" on page 25
- "Patterns for Filtering Records of Type NSS, NCP, and CIFS" on page 28
- "Filter Pattern Examples" on page 38
- "Troubleshooting" on page 40

vlog(8)

Name

vlog - The OES Storage Services Auditing Client Logger utility.

Synopsis

```
/opt/novell/vigil/bin/vlog [OPTIONS]
```

For information about options, see "VLOG Options" on page 21.

Availability

Open Enterprise Server

Syntax

Prior to running `vlog`, the NSS Auditing Engine (`/usr/lib/systemd/system/novell-vigil`.service) should be started. To check the status of the engine, issue the following command (as the `root` user) in a terminal console:

```
systemctl status novell-vigil.service
```

NOTE: After starting vigil, the service status is displayed as active (exited) instead of active (running) as compared to other services status, which displays as active (running). This is acceptable in case of vigil.

If the status is not "Running", the engine should be started by issuing the following command (as the `root` user) in a terminal console:

```
systemctl start novell-vigil.service
```

Run the NSS Auditing Client Logger (`vlog`) utility in a terminal console (generally as the `root` user).

```
/opt/novell/vigil/bin/vlog [OPTIONS]
```

Stopping `vlog` requires a `SIGTERM` signal. This can be done by issuing a Ctrl+C in the terminal where `vlog` is running, or by using the `kill` or `killall` command. For example, to kill all instances of `vlog`, enter the following in a terminal console:

```
killall -s SIGTERM vlog
```

IMPORTANT: If `vlog` terminates because of a `SIGTERM` signal, it instructs the NSS Auditing Engine (`vigil`) to discontinue sending auditing data records to the specific instance of `vlog`.

If the application is terminated without a `SIGTERM` signal (such as closing the window in which `vlog` is running), the NSS Auditing Engine does not discontinue sending auditing records to the `vlog` instance. Because an auditing record log entry is sent by appending it to the log file instance, the log file continues to grow. Over time, the audit log files can grow to fill all available disk space. See "Troubleshooting" on page 40 for further details.

Description

Intercepts, parses, filters, augments, and displays auditing records received from the NSS Auditing Engine (`vigil`).

VLOG Options

By default, `vlog` sends its output to `stdout` in an XML record format. The following options modify `vlog`'s default behavior:

`[-a, --asciiOut]`

Prints Unicode 16 (NSS paths) in `\uXXXX` format. Without this option, Unicode 16 is output in UTF-8 format.

`[-b, --blockNssEventsOfVol] NSS-VOLUME-NAME`

Blocks events (of type NSS) from the specified NSS volume from audit record stream. This option is followed by exactly one NSS volume name. This option can be specified multiple times on the command-line to specify multiple NSS volumes.

`[-c, --clientKey] CLIENT-KEY`

Sets the NSS Auditing client key to `CLIENT-KEY`. If this option is not specified, the default client key of "`Zarahemla`" is used. The `CLIENT-KEY` value is required to attach a new instance of `vlog` to a pre-existing NSS Auditing client data stream.

`[-C, --clientName] CLIENT-NAME`

Sets the NSS Auditing client name to `CLIENT-NAME`. The `CLIENT-NAME` value length is limited to 1 to 15 bytes long, and must be unique on the system.

If this option is not specified, `vlog` generates a unique, random NSS Auditing client name (and is not limited to the 1 to 15-byte-length limit of this option). The `CLIENT-NAME` value is required to attach a new instance of `vlog` to a pre-existing NSS Auditing client data stream.

`[-d, --daemonize]`

Uses daemon mode, which causes `vlog` to run in the background. All output directed to `stdout` or `stderr` is eliminated.

[-e, --extension]

Specify a new extension for the log file that was renamed (`-R` option). If no extension is specified, it will continue with the extension of the original file.

`[-f, --format] [NUL, XML, CSV, SENT, CEF]`

Sets the output format using the specified format as follows:

NUL

No output.

XML

(Default) Extensible Markup Language (XML) format.

CSV

Comma Separated Values (CSV) format.

SENT

Format compatible with NetIQ Sentinel/Log Manager products.

CEF

Common Event Format (CEF).

[-F, --filterFile] FILE-PATH

Specifies a path to a filter file that contains filter patterns to be applied to the auditing records that are received from the NSS Auditing Engine (`vigil`). This option can be specified multiple times on the command-line to specify multiple filter file paths. See "Filtering Records" on page 25 for further details.

[-h, --help]

Causes `vlog` to display a help page, then terminate.

[-k, --keepLogFiles]

Keeps unprocessed log files upon program termination. Log files represent the auditing data stream between the NSS Auditing Engine and `vlog`. The `vlog` files are stored in directory

`/var/log/audit/vlog/{clientName}/`

If this option is not specified, `vlog` destroys this directory, along with all of its content (including the cursor file, if it is implemented), when `vlog` terminates.

[-K, --keepClient]

Sets the NSS Auditing client flag `VIGIL_CLIENT_F_KEEP`, which causes `vlog` to leave the NSS Auditing client data stream open upon normal termination, and creates an orphaned NSS Auditing client data stream. If this option is not specified, the `VIGIL_CLIENTF_KEEP` flag is not set, causing the NSS Auditing client data stream to close upon normal `vlog` termination.

[-l, --size] SIZE-IN-BYTES

Limits the NSS Auditing client's audit log file to the `SIZE-IN-BYTES` (before rolling to a new file).

Specify the maximum log file size in bytes. If the specified value is not a multiple of 4096 bytes (4 KB), it is automatically rounded up to the nearest 4-KB block. By default, the log file size is 262144 bytes (256 KB). The minimum log file size is 36864 bytes (36 KB). A `SIZE-IN-BYTES` value of zero [0] indicates `no limit`.

The rollover to a new log file occurs based on space available for writes as you approach the maximum file size. Thus, the file sizes of log files might not exactly match the expected maximum size when you view their sizes in bytes. If you view their file sizes in kilobytes, all of the files will be the same size.

[--logFilePath] FILE-PATH

Specify the `vlog` log file path. The default log file location is `/var/log/audit`. The vlog file will be created in the specified location.

[-m, --maxFileCount] maxStreamFileCount

Limit the vigil auditing client's log files count. The default value for the maxStreamFileCount is set to zero (unlimited). If you have set a limit, after the number of stream files reaches the `maxStreamFileCount` value, the oldest audit streamfile is deleted to accommodate the newest data.

[-o, --outputFile] FILE-PATH

Redirects `vlog` output from `stdout` to the specified `FILE-PATH`.

[-O, --outputLaf]

Causes `vlog` to send output to the Linux Auditing Framework (LAF). This option does not cause output to stop being sent to `stdout` (or as redirected by the `[-o, --outputFile]` option). Rather, this option causes output to also be sent to LAF.

[-p, --pattern] FILTER-PATTERN

Adds a filter pattern, which filters records in the NSS Auditing data stream from `vlog` output. This option is followed by exactly one filter pattern. This option can be specified multiple times on the command-line to indicate multiple filter patterns. See "Filtering Records" on page 25 for further details.

[-q, --cursorFile]

Causes `vlog` to implement a cursor file to track the current position in the NSS Auditing client data stream. The memory-mapped file contains both the path of the current audit log file being processed, as well as the offset of the audit record being processed in that file. If `vlog` unexpectedly terminates, the cursor file preserves this information, allowing a future instance of `vlog` to reconnect to the NSS Auditing client data stream, and continue processing records in the stream, in the same audit log file, and at the same record offset, where the prior (terminated) `vlog` instance had stopped. The cursor file is stored in the same directory as the audit log file(s):

`/var/log/audit/vlog/{clientName}/`

[-r, --recCnt] NUMBER-OF-RECORDS

Limits the NSS Auditing client's audit log file to `NUMBER-OF-RECORDS` records before rolling to a new file. A `NUMBER-OF-RECORDS` value of zero [0] indicates "no limit". If this option is not specified, the default value for `NUMBER-OF-RECORDS` is zero [0].

[-R, --fileRollSize] SIZE

Specify the size of a vlog audit file (in megabytes). On exceeding the size, the file rolls over to a new file and is renamed as "`filename_timestamp.log`". If the specified value is not a multiple of 4096 bytes (4 KB), it is automatically rounded up to the nearest 4 KB block.

The log file size; maximum: 100 megabytes (MB) and minimum: 256 kilobytes (KB).

For example, ./src/vlog -R 50 -o ~/debug/john/vlog_events.log

A new file "`T_<timestamp>.log`" is created after `vlog_events.log` file reaches 50 MB. For the size in KB, suffix the size with the KB unit.

[-S, --shareClient]

Causes `vlog` to set the NSS Auditing client flag: `VIGIL_CLIENT_F_SHARE`. This flag allows another instance of `vlog` to attach to the NSS Auditing data stream. Without this option, other instances of `vlog` are unable to connect the NSS Auditing data stream created by this client.

[-t, --triggerFileExtension]

Specify an extension for the trigger files. By default, the CEF format displays the extension as `cef_ready`. The other formats will not have any extension.

[-u, --userKey] USER-KEY

Sets the NSS Auditing client user key to USER-KEY. If this option is not specified, the default value of "Teancum" is used as the client user key.

[-U, --userName] USER-NAME

Sets the NSS Auditing client user name to USER-NAME. If this option is not specified, vlog generates a unique, random, NSS Auditing client user name for the vlog instance.

[-v, --incverbose]

Debugging option. Increments the vlog's (error and warning) verbose level by one, and reports the verbose level. Use the -V option to specify a verbose level. Verbose messages are sent to stderr, and to the system message log.

[-V, --setverbose] VERBOSE-LEVEL

Debugging option. Sets vlog's (error and warning) verbose level to the specified value. Verbose messages are sent to stderr, and to the system message log. Each level includes the messages of the lower levels.

Verbose Levels	Description
60 or greater	vlog debugging messages.
50 or greater	Filter-parsing notes.
40 or greater	Program-warning notes (mostly due to filter pattern issues).
30 or greater	Filter-related notes (why audit records match, or are excluded, by filters).
20 or greater	Internal modes are noted (mostly cursor-file related modes).
10 or greater	Configuration changes are noted (as per command-line options).
0 or greater	Normal (default). Only fatal errors are emitted.
Less than 0	Silent. No messages are emitted.

For example, to set the verbose level to 22, enter

```
/opt/novell/vigil/bin/vlog -V 22
```

The verbose messages for fatal errors, configuration changes, and internal modes are sent to stderr.

[-x, --closeAuditClient]

Closes the specified auditing client data stream before opening (or re-opening) the NSS Auditing client data stream. Requires that the [-C, --clientName] and [-c, --clientKey] options be previously specified on the command line.

If this option is not specified, vlog first attempts to re-attach to a pre-existing NSS Audit data stream; and opens a new NSS Audit data stream if the attempt to attach to a pre-existing stream fails.

[-z, --exit]

Causes vlog to terminate immediately. This option is used internally by vlog when vlog calls itself recursively (as does the [-Z --libraryLinkage] option).

[-Z, --libraryLinkage]

Debugging option. Causes `vlog` to display `libvigil` linkage information, and terminates.

--filterTest

Filter pattern debugging option. Causes `vlog` to validate filter patterns, and terminates. Filter patterns specified with the `[-p, --pattern]` and `[-F, --filterFile]` options are validated.

Filtering Records

Auditing records can be filtered by the NSS Auditing engine (upstream), or can be filtered by `vlog` (downstream).

The `[-b, --blockNssEventsOfVol]` option is an example of filtering by the NSS Auditing engine. The benefit of this (upstream) filtering method is that a drastic reduction of data in the client's data stream can be realized.

The `vlog` application supports (downstream) filtering of events, as they are received from the NSS Auditing Engine (`vigil`), by using filter patterns. Filter patterns are rules for filtering events. You can use either of the following methods to specify filter patterns:

- A filter file of filter patterns (consisting of one filter pattern per line) can be specified with the `[-F, --filterFile]` command line option. This option must be followed by a `[path/]filename`.

  A filter file can contain comment lines. Comment lines begin with a pound sign (#) or a double forward slash (`//`).
- Individual filter patterns can be specified with the `[-p, --pattern]` command line option. This option must be followed by a quoted filter pattern.

There are two kinds of patterns that can be specified from a filter file by using the `[-F, --filterFile]` option, or specified individually in the command by using the `[-p, --pattern]` option.

- Patterns for filtering records of type VIGIL
- Patterns for filtering records of type NSS, NCP, and CIFS

Each of these pattern types are discussed below.

Patterns for Filtering Records of Type VIGIL

Records of type VIGIL represent operations internal to the NSS Auditing Engine. By default, records of type VIGIL are not filtered from `vlog`'s output.

- “START” on page 26
- “STOP” on page 26
- “NCP_START” on page 26
- “NCP_STOP” on page 26
- “NSS_START” on page 26
- “NSS_STOP” on page 27

- “CIFS_START” on page 27
- “CIFS_STOP” on page 27
- “CLIENT_START” on page 27
- “CLIENT_STOP” on page 27
- “USER_START” on page 27
- “USER_STOP” on page 27
- “ROLL” on page 27
- “ALL” on page 27

Filter Syntax for Type VIGIL Records

The general pattern for filtering records of type VIGIL is:

```
:[+ or -]KEYWORD [[+ or -]KEYWORD]
```

A pattern used to filter records of type VIGIL has a colon (`:`) as the first character of the pattern.

The colon is followed by one or more keywords that represent records that are to be included or excluded from the `vlog` output. Multiple keyword entries are separated by a space. Keywords are applied in the order that they appear in the filter pattern.

The specified keyword causes specific records of type VIGIL to be included or excluded from the output. Each keyword is preceded by an exclude/include character that indicates whether the records that match the specified pattern should be excluded or included in the `vlog` output. A minus (`-`) character indicates that the records that are represented by the keyword that follows it should be excluded from the `vlog` output. A plus (+) character indicates that the records that are represented by the keyword that follows it should be included in the `vlog` output

Filter Keywords for Type VIGIL Records

The keywords for VIGIL record types are as follows:

START

Each time the `vigil.ko` kernel module is loaded, a "Start" record is sent to all auditing clients.

STOP

Each time the `vigil.ko` kernel module is unloaded, a "Stop" record is sent to all auditing clients.

NCP_START

Each time the `vigil.ncp.ko` kernel module is loaded, an "NCP started" record is sent to all auditing clients.

NCP_STOP

Each time the `vigil.ncp.ko` kernel module is unloaded, an "NCP stopped" record is sent to all auditing clients.

NSS_START

Each time the `vigil.nss.ko` kernel module is loaded, an "NSS started" record is sent to all auditing clients.

NSS_STOP

Each time the `vigil.nss.ko` kernel module is unloaded, an "NSS stopped" record is sent to all auditing clients.

CIFS_START

Each time the `vigil.cifs.ko` kernel module is loaded, a "CIFS started" record is sent to all auditing clients.

CIFS_STOP

Each time the `vigil.cifs.ko` kernel module is unloaded, a "CIFS stopped" record is sent to all auditing clients.

CLIENT_START

Each time a new Auditing Client (an internal NSS Auditing Engine construct) is activated, a "Client started" record is sent to all auditing clients.

CLIENT_STOP

Each time a new Auditing Client (an internal NSS Auditing Engine construct) is deactivated, a "Client stopped" record is sent to all auditing clients.

USER_START

Each time a new Auditing Client User (an internal NSS Auditing Engine construct) is activated, a "User started" record is sent to all auditing clients.

USER_STOP

Each time a new Auditing Client User (an internal NSS Auditing Engine construct) is deactivated, a "User stopped" record is sent to all auditing clients.

ROLL

The NSS Auditing Engine (`vigil`) appends auditing records to a file in a directory specified by the auditing client application. The auditing client application can also specify the maximum size of the file in which auditing records are placed, and can optionally specify that when the file maximum size has been reached, the NSS Auditing Engine creates a new file (in the specified directory) and begins appending audit records to the new file.

When the NSS Auditing Engine creates a new file (in which audit records will be placed), it generates a "Roll" audit record, and appends it (as the last record) to the previously used file. The record contains the full (Linux) path of the newly created file, where audit record processing should continue. Roll records are sent only to the specifically affected auditing client, not to all auditing clients.

ALL

Used to indicate all records of type VIGIL.

Examples for Filtering VIGIL Events

The following are examples of how records of type VIGIL might be filtered from the `vlog` output by specifying individual patterns at the command line prompt:

`/opt/novell/vigil/bin/vlog -p":-all"`

Specifies a filter pattern that excludes all records of type VIGIL from the `vlog` output.

/opt/novell/vigil/bin/vlog -p":-all +roll"

Specifies a filter pattern that excludes all records of type VIGIL from the `vlog` output, except `Roll` records, which are shown in the `vlog` output.

/opt/novell/vigil/bin/vlog -p":-roll -user_stop -user_start"

Specifies a filter pattern that excludes `Roll` records, `User stopped` records, and `User started` records from the `vlog` output.

Keywords are applied in the order that they appear in the filter pattern. For example, the following patterns are not equivalent:

/opt/novell/vigil/bin/vlog -p":-all +roll"

Specifies a filter pattern that excludes all records of type VIGIL, but then allows the `Roll` record. Of all the VIGIL type records, only the `Roll` events are output.

/opt/novell/vigil/bin/vlog -p":+roll -all"

Specifies a filter pattern that allows the `Roll` record, but then excludes all records of type VIGIL. No VIGIL type records (of any event type) are output.

Patterns for Filtering Records of Type NSS, NCP, and CIFS

Records of type NSS, NCP, and CIFS represent operations on files.

- "! (exclamation mark character)" on page 29
- "- (minus character)" on page 29
- "+ (plus character)" on page 29

Filter Syntax for Type NSS, NCP, and CIFS Records

`[`*`negation_element`*`]`*`path_element`* `(event [event...])`

Patterns for filtering records of type NSS, NCP and CIFS consist of three elements in the following order:

1. **Negation Element:** Indicates whether records that match the specified filter patterns that follow are to be included or excluded from the auditing log. The negation element, if present, is immediately followed by the path element.
2. **Path Element:** A filename-matching pattern or directory-name-matching pattern that specifies directories or files to include or exclude from the audit log. The path element is delimited from the event element by a [space or tab] character.
3. **Event Element:** A list of NSS file system events enclosed in parenthesis to include or exclude from the audit log.
4. **User Element:** The user element of the filter pattern is a user name matching pattern that specifies users to include or exclude from the audit log. This is the userdn (.CN=ediruser.O=org.T=Tree.) for eDirectory user and netbios name (domainnameser) for the Active directory user.
5. **Application Element:** The application element of the filter pattern is a binary name matching pattern that specifies binary names to include or exclude from the audit log. This is the binary name of a application.

The options for each of the elements is described below.

Negation Element Options

The negation element is a single character that is used to indicate whether the specified filter patterns are to be included or excluded from the auditing log.

The negation element is a single character that precedes the path element:

! (exclamation mark character)

Used to negate the filter patterns specified in a filter file when the command line filter file option `[-F, --filterFile]` is used.

- (minus character)

Used to exclude (negate) a filter pattern that is specified on the command line when the filter pattern option `[-p, --pattern]` is used.

+ (plus character)

Used to include (non-negate) a filter pattern that is specified on the command line when the filter pattern option `[-p, --pattern]` is used.

The negation element, if present, is immediately followed by the path element.

Audit records that match a negated filter pattern are excluded from the `vlog` output. Specifically, `vlog` uses the following logic to include and exclude audit records from being output:

1. If there are no include (non-negated) filter patterns specified (either in the filter pattern `[-p, --pattern]` option or the filter file `[-F, --filterFile]` option), include all audit records in the output.
2. If one or more filter patterns are specified, place the include pattern before the exclude pattern, such as:

   ```
   DATA:/**ProjectX/** (*)
   !DATA:/**ProjectX/**/*.tmp (*)
   ```

   a. If the audit record does not match any of the include (non-negated) filter patterns, do not output the record.

   b. If the audit record matches any of the include (non-negated) filter patterns in 2a above, and if the audit record matches any of the exclude (negated) filter patterns that follow the include, do not output the record.

Path Element Options

The path element of the filter pattern is a filename-matching pattern or directory-name-matching pattern that specifies directories or files to include or exclude from the audit log.

The path element immediately follows the negation element (if present).

The path element is delimited from event element by a [space or tab] character. Thus, if a path element contains a space or tab character, you must do one of the following: Enclose the path element in quotation marks (such as "VOL1:/xyz dir/"), or escape each space or tab character within the path element by preceding it with a backslash (\) character (such as VOL1:/xyz\ dir/).The event element follows the path element delimiter (space character or tab character).

- "Path Element Wildcard Characters" on page 30
- "Path Element Examples" on page 30

Path Element Wildcard Characters

The syntax for the filename and directory name pattern allows for the following wildcard characters:

?

Using the question mark (?) wildcard matches any single character, except for a forward slash character (/).

*****

Using the single asterisk (*) wildcard matches any sequence of zero or more characters, except for a forward slash character (/).

******

Using the double asterisk (**) wildcard matches any sequence of zero or more characters, including a forward slash character (/).

[chars]

Using the [chars] wildcard matches any single character in chars. If chars contains a sequence of the form a-b, then any character between a and b (inclusive) will match. The forward slash (/) cannot be specified within chars. A leading or trailing minus character (-) simply includes the minus as one of the characters in the group.

\x

Using the backslash before a character matches the character specified, except for the forward slash character (/).

{a,b,...}

Using the list of strings matches any of the strings a, b, and so on.

NOTE: Generally, the forward slash character (/) must be matched explicitly. The only exception is in the use of a double asterisk (**).

For path element examples, see "Path Element Examples" on page 30.

Path Element Examples

This section provides examples of path elements and a description of how each might be applied for file path examples. If the file path does not match the path element, an explanation is provided.

/a[-e]?/joke

/a-h/joke

Match.

/adh/joke

No match. The [e-] group includes only "e" and "-", not "d".

/aeh/joke

Match.

/a[c\-e]?/joke

/a-h/joke

Match.

/ach/joke

Match.

/adh/joke

No match. The [c\-e] group does not include "d".

/aeh/joke

Match.

/a[d-f]?/joke

/aez/joke

Match.

/agf/joke

No match. Need a character from the [d-f] group.

/a[d-fs-u]?/joke

/aef/joke

Match.

/aft/joke

Match.

/a[def]?/joke

/ad/joke

No match. No character matches the "?".

/aef/joke

Match.

/agf/joke

Match.

/a[def][hij]?/joke

/afh/joke

No match. No character matches the "?".

/afhz/joke

Match.

/agfh/joke

No match. Need character from the [def] group.

/a[e-]?/joke

/a-h/joke

Match.

/aeh/joke

Match.

/afh/joke

No match. The [e-] group includes only "e" and "-", not "f".

/a*

/a/b/c/d

No match. The "*" does not match the "/" character.

/a*?/a

/a/a

No match. The "?" in the pattern does not match "/".

/ab/a

Match.

/abb/a

Match.

/a*?\/a

/ab/a

No match. An escaped "/" in the pattern cannot match "/".

/a**

/a/b/c/d/

Match.

/a?/a**

/a/a

No match. No character matches the "?".

/a/b/c/a

Match.

/a/**

/a/b/c/d

No match. Must end with the "/" character.

/a/b/c/d/

Match.

/a*/

/a/b/c/d

No match. The "*" does not match the "/" character.

/a/*/b/c//e/f**

/a/b/b/c/d/d/d/d/e/f

Match.

/a/b/b/c/d/d/d/d/e/f/e/f/e/f

Match.

/a/b/b/c/d/d/d/d/e/f/

No match. Must end with the "f" character.

/a/b/c/d/e/e/e/e/f

No match. Need something between a/ and /b/c.

/a/xxx/b/d/e/e/e/e/f

No match. a/xxx/b must be followed by /c.

/a/*/b/c//e/f/**

/a/b/b/c/d/d/d/d/e/f

No match. Must end with the "/" character.

/a/*/d/e

/a/d/e/

No match. Must end with the "e" character.

/a/def/d/e

Match.

/a/def/d/e/

No match. Must end with the "e" character.

abc,{,,,{,x,,},,,}def

abc,def

Match.

abc,}{,,,{,x,,},,,}}def

abc,}}def

Match.

abc{,,,{,x,,},,,}def

abcdef

Match.

abcxdef

Match.

abcydef

No match. Nothing after "abc" allows "y".

abc{}def

abcdef

Match.

abc*{def,xyz,hij,{a*[d-g],b*[7-9]}z}qrt*m

abcadzqrtm

Match.

abcaezqrtm

Match.

abcb6zqrtm

No match. Nothing after "abcd" would allow "6".

abcb8zqrtm

Match.

abcdefadzqrtm

Match.

abcdefdefqrtm

Match.

abcdefqrtm

Match.

abchijqrtm

Match.

abcxyqrtm

No match. Nothing after "abc" would allow "xyq".

abcxyzdefqrtm

Match.

abcxyzqrtm

Match.

For simplicity, the NSS volume name was omitted from the above examples. NSS audit records emitted by the NSS Auditing Engine (`vigil`) have paths that include the NSS volume name. In order to match these NSS audit records, the NSS volume name must be specified as part of the path element, similar to the following:

VOL1:/a*?/a

VOL1:/ab/a

Match.

VOL2:/a*?/a

VOL1:/abb/a

No match.

VOL2:/a?/a**

VOL2:/a/b/c/a

Match

User Element Options

The user element of the filter pattern is a user name matching pattern that specifies users to include or exclude from the audit log. This is the userdn (.CN=ediruser.O=org.T=Tree.) for eDirectory user and netbios name(domainnameser) for the Active directory user.

The user element is delimited by a [comma] character. The user element follows the event element. It is mandatory to specify the path and event pattern before the user element.

The asterisk character (*) can be used to specify all users.

The exclamation mark (!) or minus (-) characters can precede a user to exclude it.

NOTE: If the application name or user name contains a comma ',' or closing bracket ')', ensure to prefix it with the escape character '\'. If the escape character is not used, the behavior of vlog filter is undefined.

User Element Examples

This section provides examples of user element patterns and a description of how each might be applied.

`(*)`

Matches all the users.

`(!.CN=Joe.O=novell.T=Tree.)`

Matches all users except Joe.

`(!ADdomain\Joe)`

Matches all users except the Active Directory user Joe.

Application Element Options

The application element of the filter pattern is a binary name matching pattern that specifies binary names to include or exclude from the audit log. This is the binary name of a application.

The application element is delimited by a [comma] character. The application element follows the user element. It is mandatory to specify the path, event, and user pattern before the application element.

The asterisk character (*) can be used to specify all applications.

The exclamation mark (!) or minus (-) characters can precede a application name to exclude it.

NOTE: If the application name or user name contains a comma ',' or closing bracket ')', ensure to prefix it with the escape character '\'. If the escape character is not used, the behavior of vlog filter is undefined.

Application Element Examples

This section provides examples of application element patterns and a description of how each might be applied.

(*)

Matches all the applications.

(!nbackup)

Matches all applications except nbackup.

Event Element Options

The event element consists of a list of events enclosed in parentheses. The events listed in the parentheses are delimited by a [space or tab] character.

The event element follows the path element delimiter (space character or tab character).

- "Event Types" on page 36
- "Event Type Examples" on page 37
- "Event Sub-Types NSS, NCP, and CIFS" on page 37
- "Event Sub-Type Examples" on page 38

Event Types

Valid event options are:

```
DELETE
CREATE
OPEN
CLOSE
RENAME
MODIFY
ADDTRUSTEE
REMOVETRUSTEE
SETINHERITEDRIGHTS
```
LINK (refers to hard links to files)

IMPORTANT: NSS and NCP Server support hard links to files but not to directories. Hard links to files are supported only for paths within the same NSS volume. If you create a hard link, it is reported as a LINK event.

Because of security considerations, NCP Server intentionally does not support soft links. If you create a soft link, it is reported as a CREATE event with a file attribute of 21-IS_LINK.

The asterisk character (*) can be used to specify all events.

The exclamation mark (!) or minus (-) characters can precede an event name to exclude (negate) events of that type.

The event options can occur in any order in the parenthesis. When parsing an event list, a list of included (non-negated) events is first created, then the excluded (negated) events (that is, those with the [! or -] exclusion character in front of them) are removed from that list.

Beginning in OES 2015, a delete action for a file or folder on the NSS file system from an NCP client or CIFS client triggers four events: DELETE, OPEN, MODIFY, and CLOSE. In OES 11 SP1 and earlier, a delete action triggers only a DELETE event.

Event Type Examples

The following are examples of event element pattens and a description of the results you can expect for each one:

(*)

Includes all elements.

(OPEN CLOSE RENAME)

Includes only the OPEN, CLOSE, and RENAME events.

(* !OPEN)

Includes all events except OPEN.

This list could also have been specified as (!OPEN *). All excluded (negated) events are removed after first creating a list of events that are included (non-negated).

(OPEN CLOSE !RENAME)

Includes only the OPEN and CLOSE events.

The !RENAME is not necessary here because it was never included. Typically, you add the excluded events in the list only when you use the asterisk (*) to specify all events.

(!* OPEN CLOSE)

Excludes all events. No events are included because all events were excluded.

This example illustrates that you can put the [! or -]character in front of the asterisk (*).

The OPEN and CLOSE event names have no effect because all excluded (negated) events are processed after the included (non-negated) events. In effect, this list includes the OPEN and CLOSE events, then excludes all events.

(OPEN !OPEN CLOSE)

Includes only the CLOSE event.

The OPEN and !OPEN cancel each other out.

Event Sub-Types NSS, NCP, and CIFS

Each event in the list can also specify a list of event sub-types to include or exclude. These are specified in parentheses immediately after the operation name. Event sub-types are delimited by a space or tab.

Valid event sub-types are:

NSS
NCP
CIFS

The asterisk (*) can be used to include all event sub-types.If no sub-types are specified, they are all included by default.

The exclamation mark (`!`) or minus (`-`) characters can precede a sub-type to exclude it.

The sub-types can occur in any order in the parentheses. When parsing a sub-type list, a list of included (non-negated) sub-types is created first, then the excluded (negated) sub-types (that is, those with the `[! or -]` exclusion character in front of them) are removed from that list.

Event Sub-Type Examples

This section provides examples of event element patterns and a description of how each might be applied.

`(*)`

Matches all events.

`(OPEN)`

Matches all OPEN events (including the NSS, NCP, and CIFS sub-types).

`(OPEN CLOSE)`

Matches only the OPEN and CLOSE events (including the NSS, NCP, and CIFS sub-types).

`(OPEN(NSS) CLOSE(NSS))`

Matches only the NSS OPEN and NSS CLOSE events.

`(* !OPEN)`

Matches all events (including the NSS, NCP, and CIFS sub-types) except the OPEN event.

`(* !OPEN(NSS) !CLOSE(NSS))`

Matches all events except the NSS OPEN and NSS CLOSE events.

`(* !OPEN(* !NSS) !CLOSE(* !NSS))`

Matches all events except the non-NSS OPEN and non-NSS CLOSE events.

Filter Pattern Examples

This section provides examples of filter patterns and a description of how each might be applied. These examples are specific to entries in the filter file (when using the `[-F, --filterFile]` option).

`VOL1:/abc/* (*)`

Include: Matches all events on any file in the `VOL1:/abc` directory. The events on files in subdirectories are not included. It will not match the directory “abc”.

`VOL1:/abc/** (*)`

Include: Matches all events on any file in the `VOL1:/abc` directory and any of its subdirectories. It will not match the directory “abc”.

`VOL1:/abc/* (OPEN CLOSE)`

Include: Matches only OPEN and CLOSE events on any file in the `VOL1:/abc` directory. The events on files in subdirectories are not included.

VOL1:/abc/* (* !OPEN !CLOSE)

Include: Matches all events except the OPEN and CLOSE events on any file in the VOL1:/abc directory.

VOL1:/abc/* (OPEN(NSS) CLOSE(NSS))

Include: Matches only NSS OPEN and NSS CLOSE events on any file in the VOL1:/abc directory.

VOL1:/abc/* (* !OPEN(NSS) !CLOSE(NSS))

Include: Matches all events except the NSS OPEN and NSS CLOSE events on any file in the VOL1:/abc directory.

VOL1:/abc/* (* !OPEN(* !NSS) !CLOSE(* !NSS))

Include: Matches all events except the non-NSS OPEN and non-NSS CLOSE events on any file in the VOL1:/abc directory.

!VOL1:/abc/def (*)

Exclude: Matches all events for the VOL1:/abc/def file. This exclusion causes all events on the VOL1:/abc/def file to be dropped (assuming that they had been included by an include rule).

!VOL1:/abc/def (OPEN)

Exclude: Matches OPEN events for the VOL1:/abc/def file. This exclusion effectively excludes OPEN events on the VOL1:/abc/def file.

VOL1:/abc/def (* !OPEN)

Include: Matches all events except the OPEN event on the VOL1:/abc/def file. This is another way to exclude OPEN events on the VOL1:/abc/def file.

!VOL1:/abc/def (* !OPEN)

Exclude: Matches all events except the OPEN event on the VOL1:/abc/def file. This effectively excludes all events except the OPEN event on the VOL1:/abc/def file (assuming they had been included by an include rule).

"VOL1:/xyz dir/*" (*)

Include: Matches all events on any file in the "VOL1:/xyz dir" directory. Quotation marks are used to enclose the pattern when the path contains spaces. Otherwise, the space in "xyz dir" is treated as a delimiter between the pattern and the events.

VOL1:/xyz\ dir/* (*)

Include: Matches all events on any file in the "VOL1:/xyz dir". A backslash (\) is used to escape the space character in the directory name "xyz dir". Otherwise, the space in "xyz dir" is treated as a delimiter between the pattern and the operations.

VOL1:/abc/ (*) (*)**

Include: Matches all events for all the users on "VOL1:/abc".

VOL1:/abc/ (*) (!root)**

Include: Matches all events for all the users other than "root" on "VOL1:/abc". This is a way to exclude all events by the "root" users on the "VOL1:/abc".

!VOL1:/abc/ (*) (.CN=admin.O=novell.T=VIGIL_NSSAD-TREE.)**

Exclude: Matches all events by the "admin" user on the "VOL1:/abc". This effectively excludes all the events by the "admin" user on the "VOL1:/abc".

!VOL1:/abc/ (*) (.CN=tom.O=novell.T=VIGIL_NSSAD-TREE.,addomain\Joe)**

Exclude: Matches all events by the eDirectory user "tom" and Active Directory user "Joe" on the "VOL1:/abc". This effectively excludes all the events by the users "tom" and "Joe" on the "VOL1:/abc".

VOL1:/abc/ (*) (root) (!nbackup)**

Include: Matches all events for the applications other than "nbackup" for only the root user on "VOL1:/abc". This excludes all nbackup events by the "root" user on the "VOL1:/abc".

!VOL1:/abc/ (*) (.CN=admin.O=novell.T=VIGIL_NSSAD-TREE.) (nbackup)**

Exclude: Matches all events for the application nbackup by the "admin" user for the "VOL1:/abc". This excludes all the nbackup events by the "admin" user on the "VOL1:/abc".

Troubleshooting

Orphaned Auditing Client

The NSS Auditing Engine (vigil) implements an interface that allows user-space applications (such as vlog) to establish Auditing Clients. When doing so, the user-space application (such as vlog) specifies various parameters, such as a directory in the Linux file system where the auditing record log files are placed.

After an Auditing Client has been established, the NSS Auditing Engine (vigil) architecture has been designed to store the auditing records in files in the directory specified by the auditing application (such as vlog or Novell Sentinel).

Records are stored in the files until the Auditing Engine is instructed to stop, or until the NSS Auditing Engine is stopped. If the auditing application terminates (perhaps unexpectedly), and the NSS Auditing Engine is therefore not instructed to stop sending records to the Auditing Client's directory, the NSS Auditing Engine continues to store auditing records in the Auditing Client's specified directory.

An Auditing Client that does not have a live user-space application associated with it is called an Orphaned Auditing Client. The architecture of the NSS Auditing Engine supports this mode of operation. This mode facilitates the continued collection of auditing data, even if the auditing application temporarily fails. The NSS Auditing Engine architecture assumes that the auditing application will eventually be restarted, and will then re-connect to the auditing stream.

The default configuration of the vlog application does not attempt to re-connect to an orphaned Auditing Client from a previous failed vlog session. If vlog is not properly terminated by the SIGTERM signal, an Orphaned Auditing Client is created.

IMPORTANT: If Orphaned Auditing Clients are not stopped, they continue until they fill the Linux file system partition with auditing data.

You can use one of the following methods to eliminate Orphaned Auditing Clients: Start and stop (or restart) the NSS Auditing Engine, or stop a specific instance of the Auditing Client. Each method is described below.

Method 1: Stop and Start (or Restart) the NSS Auditing Engine

To do this, enter the following commands as the `root` user at a terminal console prompt:

```
systemctl stop novell-vigil.service
```

```
systemctl start novell-vigil.service
```

Or you can enter the following command to restart the engine:

```
systemctl restart novell-vigil.service
```

This method stops all Auditing Clients, including those that were not associated with the `vlog` application. This might be undesirable because some auditing records of file-system events will not be logged to the various auditing applications.

Method 2: Stop a Specific Auditing Client Instance

By default, all active Auditing Clients for the NSS Auditing Engine can be listed by listing the directory content of the `/sys/audit/vigil` directory.

For example, enter the following command as the `root` user at a terminal console prompt:

```
ll /sys/audit/vigil
```

All active Auditing Clients are represented in the listing as directories named “`CLIENT_*`”. Using the `[-C, --clientName]` option, `vlog` Auditing Clients can be given a name such as “`JOHN`”, and the specific entry in the `/sys/audit/vigil/` directory will be “`CLIENT_JOHN`”. If the `[-C,--clientName]` option is not specified, `vlog` generates a random Auditing Client name. Generated name entries are prefixed with "`CLIENT_VLOG_`", followed by the process ID that created the client, followed by a numeric value that represents the date and time that the specific Auditing Client was started.

For example, a `vlog` generated client name might be:

```
CLIENT_VLOG_31691-1264200226
```

In order to stop a specific instance of a `vlog` Orphaned Auditing Client, that client's entry in `/sys/audit/vigil/` must be accurately identified. The `root` user can stop a specific instance of an auditing client by writing a `STOP` command to the client's `CONTROL` file (found in that client's directory). The `ClientKey` must also be specified as an additional credential. This limits closing an Auditing Client to a `root` user who knows the Auditing Client's `ClientKey`. The `ClientKey` can be specified by using `vlog`'s `[-c, --clientKey]` option. If the `[-c, --clientKey]` option is not specified, `vlog` uses the default client key "`Zarahemla`".

For example, enter the following command as the `root` user at a terminal console prompt:

```
echo 'CLOSE ClientKey="Zarahemla"' > \
>  /sys/audit/vigil/CLIENT_VLOG_31691-1264200226/CONTROL
```

IMPORTANT: When an Auditing Client is closed, the client's directory is removed. You should not close an Auditing Client if the client directory is the current working directory.

7 Common Event Format (CEF)

With OES 2018 SP2 or later, the Storage Services Auditing Client Logger (VLOG) supports output in Common Event Format. This output can be integrated with third-party auditing software that supports CEF.

The following table displays the CEF key names and their description.

Table 7-1 *CEF Key Names with Description*

CEF Key Name	Description
deviceCustomDate2	Access time of the file
not mapping	Application registered for events
sourceProcessName	Users who are registered for the application
deviceCustomNumber1	File close with flags such as delete on close
destinationProcessName	Process name who access the files
deviceCustomNumber3	File access protocol connection ID
DeviceCustomNumber2	File creation or open mode/flags
fileCreateTime	Time when the file was created or time when the file was created from protocols.
filePermission	Access rights request to open a file by protocols or File permission to rename the flags or rights permission request for the file.
destinationUserName	Users FQDN
deviceEventClassId, name	Actual file operations event from NSS, NCP, CIFS, and VIGIL (vigil events)
deviceCustomeNumber4	File handle state during the file close (such as modify, snapshot and so on)
deviceCustomNumber1	Flag to indicate delete a file on close
fileSize	Size of the file
fileType	The type of the file such as datastream and socket
deviceCustomString2	Linux file system user ID
deviceCustomString1	Linux file system user name
deviceCustomDate2	Last accessed time of a file through NCP

CEF Key Name	Description
fileModificationTime	Time when the file was last modified through NCP
filePath	Full path to the file or NSS file path or data target of the file or Rename - destination path of the file
fileModificationTime	Data modified time of deleted time of a file
deviceCustomString5	File modifier GUID
deviceCustomString6	File modifier DN
message	Modifying the file mask
sourceAddress	File access client IP address
eventOutcome	Operation return status or data status output
deviceCustomString3	File owner GUID
deviceCustomString4	File owner DN
deviceProcessId	Process PID that performs the operation
DeviceCustomNumber2	File Sgid's GUID
flexString2	File Sgid name or file Sgid (folder) name
deviceCustomNumber3	File Suid's GUID
flexString1	File Suid name or file Suid (folder) name
flexnumber1	File TaskID's GUID
deviceReceiptTime	Time of the event occurred
deviceEventCategory (only NSS info)	Data type or application type
sourceUserId	File UID's GUID
sourceUserName	File UID name or file UID (folder) name
destinationUserName	File user DN
destinationUserId	Data userid suid of suid
fileId	Data ZID ID of the file
flexnumber2	File key's GUID
oldFilePath	Data source path of old file
filePath	Data target path of the file
oldFilePermission	Old access rights for the file
The following CEF key names are OES specific attributes and hence they are preceded with OES.	
OESEgid	Linux Effective group ID

CEF Key Name	Description
OESEgidName	Linux Effective group name
OESEuid	Linux Effective user ID
OESEuidName	Linux Effective user name
OESFileAttributes	File attributes such as archive, hidden, and system while open, close, and modify
OESFileAttributesModMask	Modifying file attributes MASK
OESParentFileId	Parent file (folder) ZID
OESFileHandle	Virtual file handle for the file opened
OESRetOpenCreateAction	Operation return status for file create
OESSearchAttributes	File search (folder) attributes
OESMetaDataModified	The metadata modified time of a file
OESFileNameType	The name formats are Long, UNIX, and DOS
OESVolumeDn	FQDN of the data volume
OESVolumeId	Data volid ID of the device
OESVigilRecNo	vigilrec no id of the file
OESvlogRecNo	vlogrec no id of the file
OESFsgid	Linux file system group ID
OESFsgid_Name	Linux file system group name
OESFsguid_Name	Linux file system group name
OESGid	Linux group ID
OESGidName	Linux group name
OESGidName	Linux group name
OESPurgedFileFlag	OES specific attributes
OESFileExectueType	File execute type
OESElementType	Element type
OESPrimaryNameSpaceID	namespace used when the file was created
OESFinderInfo	Macintosh FInfo data (as stored and retrieved for Macintosh files)
OESProDOSInfo	Macintosh proDOSInfo as a 2-byte file type and 4-byte aux type for Pro DOS workstations
OESFiller	Is Unused
OESDirRightsMask	Is Unused
OESFMode	UNIX File Permission / Access Modes

CEF Key Name	Description
OESRdev	UNIX root device
OESMyFlags	Unix NS specific flag
OESNfsUID	Unix NS specific flag
OESNfsGID	Unix NS specific flag
OESNwUID	Unix NS specific flag
OESNwGID	Unix NS specific flag
OESNwEveryone	Unix NS specific flag
OESNwUIDRights	Unix NS specific flag
OESNwGIDRights	Unix NS specific flag
OESNwEveryoneRights	Unix NS specific flag
OESAcsFlags	Unix NS specific flag
OESFirstCreated	Unix NS specific flag
OESVariableSize	Additional data space size
OESVariableData	Additional data space
OESExtAttrUserFlags	An arbitrary value that is set by the user. This field only applies to extend attributes and has no particular significance to the file system
OESPoolFeaturesEnabled	Enabled pool features
OESVolFeaturesEnableModMask	Bit mask that defines which bits in the volume features are to be set and/or cleared
OESVolNdsObjectId	eDirectory volume object GUID
OESVolNdsObjectIdDn	eDirectory volume object DN
OESVolSalvageMaxKeepSeconds	The number of seconds a file must remain in a salvageable state before the file system is allowed to automatically purge the file (if free space is needed)
OESVolSalvageLowWaterMark	Low water mark percentage for the volume
OESVolSalvageHighWaterMark	High water mark percentage for the volume
OESPoolFeaturesEnabled	Enabled pool features
OESPoolFeaturesEnableModMask	Bit mask that defines which bits in the pool features are to be set and/or cleared
OESPoolNdsObjectId	Pool eDirectory object ID
OESVolDataShreddingCount	Volume data shredding count
OESVolTotalSpaceQuota	Volume total space quota
OESDirQuota	Quota information for a directory

CEF Key Name	Description
OESReadAheadBlocks	Readahead blocks
OESNumOfTrustees	Number of Trustees getting modified
OESMetaDataModifier	Metadata Modifier GUID
OESMetaDataModifierDn	Metadata Modifier Dn
OESArchived	Archived time
OESLinkFlags	Link flags
OESCreateAndOpen	Create and Open flag
OESCreateFlags	Specifies the actions to take place if the file object being created already exist
OESDesiredAccessRights	Desired Access Rights
OESNSSFileAttributes	Specifies a bit mask that identifies specific file attributes to be associated with the newly created file
OESfilePermission	Bit mask that defines which bits in the fileAttributes are to be set and/or cleared
OESLinuxPosixFileHandle	Linux Posix file handle
OESParentZid	ZID of the parent that was used to open the file.
OESRenameFlags	Bit mask that identifies various modes to the rename function
OESRequestedRights	Rights that are requested for this instance
OESSuid	SUID
OESSuidName	SUID Name
OESPmdNcpTaskID	NCP task ID
OESUid	Linux UID
OESUidName	Linux user name
OESUserDN	DN of the user performing the operation
OESUserID	GUID of the user performing the operation